def precedence(op):

if op in ('+', '-'):

return 1

if op in ('\*', '/'):

return 2

if op == '^':

return 3

return 0

def is\_operand(ch):

return ch.isalnum()

def infix\_to\_postfix(expression):

stack = []

output = ""

for ch in expression:

if is\_operand(ch):

output += ch

elif ch in "([{":

stack.append(ch)

elif ch in ")]}":

while stack and stack[-1] not in "([{":

output += stack.pop()

stack.pop()

else:

while stack and precedence(stack[-1]) >= precedence(ch):

output += stack.pop()

stack.append(ch)

while stack:

output += stack.pop()

return output

expr = input("Enter an infix expression: ")

print("Postfix Expression:", infix\_to\_postfix(expr))

Output:

Enter an infix expression: (a+(b\*c)/d)

Postfix Expression: abc\*d/+

Enter an infix expression: (2+3)\*(4^2-5)/(6+1)

Postfix Expression: 23+42^5-\*61+/